Before the
Federal Communications Commission
Washington, D.C. 20554

In the Matter of)
)
Notice Regarding Software Defined Radio) ET Docket No. 00-47
)
)

COMMENTS OF MOTOROLA

Richard C. Barth
Vice President and Director
Telecommunications Strategy
and Regulation
Motorola, Inc.
1350 I Street, N.W., Ste 400
Washington, DC 20005

John F. Lyons
Director, Telecommunications
Strategy and Regulation
Motorola, Inc.
1350 I Street, N.W., Ste 400
Washington, DC 20005

March 19, 2001

TABLE OF CONTENTS

SUMMARY

Motorola commends the Commission for issuing this *Notice* on Software Defined Radio (SDR). Motorola has been a leader in the development of SDR technologies, and we are pleased to share our knowledge of these technologies, and our recommendations regarding the proposed rule changes, with the Commission.

Motorola has focused our response commentary on four major topics: definition of Software Defined Radio; class III permissive change, third party software changes; and security. We see opportunities to clarify the definition of SDR, and at the same time, remove some ambiguity regarding the Class III permissive change rule. We also see potential administrative problems associated with third party software changes. Our discussion of security goes into considerable detail. Our intention is to provide the Commission with a substantive appreciation for the methods that Motorola, and other equipment manufacturers, will use to (in the words of the Commission) "*ensure that only software that is part of a hardware/software combination approved by the Commission or a TCB can be loaded into a radio. The software must not allow the user to operate the radio with frequencies, output power, modulation types or other parameters outside of those that were approved.*"

To aid the Commission in considering Motorola's comments, the following list summarizes the key messages, and recommendations, made throughout this response. For each point, the section from which the message, or recommendation, is taken is indicated in parenthesis.

1. **Key Message**: SDR technologies are present in both current generation base stations, and handsets; due to greater flexibility enabled by SDR in radio

products, the adoption of SDR technologies will increase with time. Nevertheless, handheld products that are optimized for specific air interfaces, or specific combinations of air interfaces, will remain a dominant percentage of the total market, up to and beyond the next five years. (Section 2.)

2. **Recommendation:** The definition for Software Defined Radio should be revised to recognize software changes which affect both desired, and undesired emissions, and to permit hardware changes which do not affect either desired, or undesired emissions. An additional definition for Software Defined Radio Technology should be added. (Section 3.3.)
3. **Key Message**: Equipment authorizations must apply to hardware-software combinations, in order to ensure compliance with emission and safety requirements. (Section 4.1.)
4. **Recommendation**: There should be no requirement to declare that a radio is an SDR at the time of original equipment authorization. The Class III permissive change rule should apply to all authorized equipment. (Section 5.)
5. **Key Message**: The creation of the Class III permissive change rule will encourage the continued, unencumbered, emergence of Software Defined Radio technology. (Section 6.1)
6. **Recommendation**: A Class III permissive should be allowed when both software and hardware are changed concurrently, so long as the hardware change does not directly degrade desired, or increase undesired, emissions. (Section 6.2)
7. **Recommendation**: A Class III permissive change should be allowed when

either a Class I, or Class II, permissive change has previously occurred. (Section 6.2)

8. **Recommendation**: The rules should clearly state that a Class II permissive change is required only when undesired emissions are degraded, not when they are simply affected. (Section 6.3)

9. **Recommendation**: The rules should clearly state that a Class II permissive change applies equally to hardware and software changes. (Section 6.3)

10. **Recommendation**: For clarity, the rules should summarize the authorization requirements, for various circumstances, in a table. (Section 6.4)

11. **Recommendation:** The submission of a software listing, at the time of equipment authorization, should not be required. (Section 6.7)

12. **Recommendation:** There should be no limit placed on the maximum number of hardware-software combinations covered under a single approval. (Section 6.7)

13. **Key Message**: Motorola encourages, and is an active participant in, the emerging, vigorous and competitive third-party software market. The software products that define this market are primarily applications that execute within secure, trusted domains, like WAP or MExE. Motorola does not envision, in the foreseeable future, a significant extension of this market to include third party software products that directly control core radio functionality. (Section 7.1)

14. **Recommendation**: Motorola supports the concept of optional electronic labeling. We recommend that electronic labeling be made optional for all

authorizations, including original grants. (Section 7.2)

15. **Recommendation**: Motorola sees the potential for significant administration problems associated with third party software changes. We recommend the creation of a joint authorization process, which would hold both the OEM, and the third party, jointly accountable for the safe and reliable operation of the hardware-software combination. (Section 7.)

16. **Key Message**: Robust security methods are essential to ensure that SDR technology does not compromise safety and interference controls. Security is ultimately the responsibility of equipment manufacturers to ensure that their products are reliable and tamper-proof. (Section 8.)

Before the
Federal Communications Commission
Washington, D.C. 20554

In the Matter of)
)
Notice Regarding Software Defined Radio) ET Docket No. 00-47
)
)

COMMENTS OF MOTOROLA

Motorola, Inc. (hereinafter Motorola) submits these comments in response to the *Notice of Proposed Rule Making* in the above-captioned proceeding.

1. Introduction

Motorola commends the Commission for issuing this *Notice* on Software Defined Radio (SDR).[1] Motorola has been a leader in the development of SDR and we are pleased to share our knowledge of these technologies, and our comments on the proposed rule changes, with the Commission. Motorola is one of the founding members of the SDR Forum, an international consortium of commercial and industrial companies. More than 100 companies, from the telecommunications industry, bring their unique expertise to this organization, which was formed to develop architecture and forge agreement on SDR standards and technology development.

[1] Authorization and Use of Software Defined Radios, Notice of Proposed Rulemaking, ET Docket No. 00-47 (rel. December 8, 2000) ("Notice" or "NPRM").

In addition to our expertise in SDR developed for national defense and public safety uses, Motorola is examining SDR potential in a number of our major business segments, including Personal Communications; Network Systems; Semiconductor Products; Commercial Government & Industrial; and Integrated Electronic Systems. These comments reflect the knowledge gained of SDR capabilities and implications from throughout this diverse set of business units.

2. State of Software Defined Radio Technology

While it is generally true that current generation base station equipment employs a greater degree of SDR technology than current generation handset equipment, and it is also true that adoption of SDR technology is increasing with time, Motorola feels that the following statement, taken from the NPRM, deserves some clarification.[2]

> *"While the technology is currently only available in base stations, widespread handset use is expected within five years."*

As discussed in our NOI response, Motorola views SDR as a collection of implementation technologies that enable greater flexibility in radio products.[3] Motorola believes that cost, size, and power dissipation constraints will force handsets to lag behind base stations in the adoption of SDR technologies. Nevertheless, most current generation commercial products, both base stations and handsets, include features which are software controlled. Some of these features are characterized by software control of a hardware subsystem. Other features are implemented directly in software, not simply under software control.

[2] *Id*. at para. 11.

Looking forward, it is reasonable to expect a continuous evolution of both base station and handset products, characterized by an ever-increasing degree of SDR technology adoption. Nevertheless, consumer demand for small, lightweight, battery efficient handsets will remain constant. This, we believe, will result in handheld products that are optimized for specific air interfaces, or specific combinations of air interfaces, to remain a dominant percentage of the total market, up to and beyond the next five years.

3. Definition of Software Defined Radio

3.1. Objectives for the Definition

Motorola sees two objectives for the FCC's establishment of a definition for Software Defined Radio. The first, and most practical, reason is to put forth the definitions that are required for unambiguous interpretation of the FCC rules. The second reason is to facilitate the general education of both industry, and the public, as to the nature of SDR technology. Given that the FCC rules center, primarily, on transmitter functionality, RF emissions, and equipment authorization, it is unlikely that one single definition will suite both of these objectives. The following paragraphs will explore issues with the current definition, and propose alternative definitions.

3.2. Issues with Current Proposed Definition

The proposed definition of SDR in the NPRM is:[4]

[3] *See* Motorola Comments, *Inquiry Regarding Software Defined Radios*, *Notice of Inquir*y, ET Docket 00-47, 15 FCC Rcd. 5930 (2000), at 3.

[4] *See NPRM* at para. 21.

"A software defined radio is a radio that includes a transmitter in which the operating parameters of the transmitter, including the frequency range, modulation type or maximum radiated or conducted output power can be altered by making a change in software without making any hardware changes."

With regard to the need to put forth the definitions required for unambiguous interpretation of the rules, Motorola sees the following potential issues with the current definition.

1. Most current generation base station and handheld products have the ability to change transmitter operating parameters through a change in software, and will, therefore, satisfy the currently proposed definition. A more meaningful distinction, which could be drawn between various current generation products, is the practicality, or ease, by which software can be changed after the product has been deployed in the field.
2. The current wording emphasizes frequency, modulation and output power, as those parameters that can be altered in an SDR, with no mention of undesired emissions, which are an essential part of the FCC rules.
3. By stating that an SDR can alter radio parameters without a hardware change, there is an implication that a device, which does require a hardware change (in order to change a radio parameter), is, therefore, not an SDR. As an example, consider a transmitter that requires the addition of memory (volatile or non-volatile) in order to process a new software revision.

With regard to the desire to educate the industry, and the public, on the nature of SDR technology, Motorola sees the following potential issues with the current definition.

1. Emphasis is placed on reconfigurable transmitter functionality, in particular, functionality that pertains to the physical channel. Motorola defines SDR more broadly. We define SDR to imply flexibility that extends from the antenna to the application environment.
2. The current definition implies that a device either is, or is not, an SDR. Whereas such a distinction may be required to facilitate some particular rules (see section 5), Motorola sees SDR as a collection of implementation technologies, which can be employed in a design to varying degrees. One might think of the use of SDR technologies in a design, in the same way one might think of the use of integrated circuits. Either could be used to a greater or lesser extent to meet particular product objectives.

3.3. Alternative Proposed Definition

Motorola offers the following definitions as examples that address the objectives, and issues, discussed in the previous paragraphs.

Software Defined Radio Technology: A collection of implementation technologies that allows the functionality of a radio to be altered by making a change in software. The functionality which can be altered by the change in software may include: modulation/demodulation, coding/decoding, link, network, transport, control and application processing.

Software Defined Radio: A radio that includes a transmitter in which the operating parameters of the transmitter can be altered by making a change in software, after the radio has been deployed in the field, and without making changes to hardware which is associated with the generation, control and amplification of the desired emissions. The functionality, which can be altered by the change in software, may include the frequency range, modulation type, radiated or conducted power, or other signal processes that affect undesired emissions.

4. General Comments on the Equipment Authorization Process

4.1. Importance of Combined Hardware and Software Authorization

Motorola agrees with the FCC's conclusion that equipment authorization must apply to hardware-software combinations.[5] We agree that this approach is necessary to insure compliance with emission requirements, and we also agree that such a policy is no more burdensome than current policies governing multimode, and multiband, devices.

To expand on this concept, it is instructive to consider the quality assurance processes that are currently employed by equipment manufacturers, and then to consider how the emergence of SDR technologies will affect these processes. For current base station, and handheld products, it is typical for a single generation of hardware to be coupled with multiple evolutionary releases of software. Equipment manufacturers insure the quality of each new hardware-software combination through a combined strategy of design, verification, and configuration control. Ultimately, manufacturers are confident that products delivered to the market place will meet all quality requirements, including those pertaining to emissions and safety. This confidence is the same, whether the software is loaded in the factory, or in the field.

With the emergence of SDR technologies, it will become increasingly common for the software to be loaded in the field (note that this is the norm for current generation cellular base stations). However, there need be no compromise to any of the aforementioned quality assurance steps: design, verification, and configuration control. Motorola believes that quality cannot, and will not, be compromised, as a result of SDR. We believe that a central requirement to achieve this quality control is that lower layer interfaces within a radio, particularly those which affect emissions and safety, must remain under tight control of the equipment manufacturer, and must be protected through robust security measures (see Section 8).

4.2. Movement Towards Self-Authorization

Motorola agrees that it is prudent, at this time, for the FCC to closely monitor the emergence of SDR technologies, and to directly administer the authorization of Software Defined Radios. We support, and encourage, an eventual movement towards the use of TCBs, and manufacturer's self-approval, once SDR technology has been adequately proven.

5. Declaration of a Device as a Software Defined Radio

The NPRM includes the following statement in regards to the declaration of a device as a SDR.[6] "*We propose that the original certification application must identify the equipment as a software-defined radio, and that only the grantee of the authorization for a software defined radio may file for a Class III permissive change.*" Such a declaration would obviously be dependent on the definition of a SDR. As discussed in Section 3, the definition of SDR, as proposed in the NPRM, is sufficiently broad that most current generation radio products would qualify as a SDR. The NPRM suggests no

[5] *Id.* at para. 18.

downside to declaring a device as a SDR, whereas the potential for a Class III permissive change is clearly beneficial. It would seem, therefore, that manufacturers would have clear motivation to declare any device, which meets the definition, as a SDR.

It is not clear, however, why it is necessary to make the distinction that a given device is a Software Defined Radio. If it is the intention of the FCC to anticipate which approved devices may, in the future, undergo a Class III permissive change, then some original declaration would be necessary. If this is not the FCC's intention, then it may be more straightforward to simply allow the Class III permissive change to apply to all radios. This would, in turn, eliminate many of the concerns over the precise definition of a Software Defined Radio (see section 3).

6. Class III Permissive Change

6.1. General Comments

Motorola applauds the FCC's efforts to create equipment authorization policy that encourages the continued, unencumbered, emergence of Software Defined Radio technology. In our response to the NOI, Motorola recommended that a completely new filing be required when a software change fundamentally altered the frequency, modulation, or output power of a radio.[7] We had recommended that software changes, which did not alter these fundamental parameters, but did affect the undesired emissions, could be approved through a streamlined filing procedure, like a Class II permissive change.

The creation of the Class III permissive change can enable an even more dynamic

[6] Id. at para. 26.

environment than we had envisioned, while at the same time, insuring that interference and safety concerns are not compromised. Motorola, therefore, endorses the Class III permissive change concept. We do have questions regarding interpretation of the proposed rule changes; these questions are outlined in the following paragraphs.

6.2. Hardware Changes and Software Changes

Motorola believes that two realistic scenarios expose certain issues and ambiguity in the following statement:[8] "*We also propose that Class III permissive changes may only be made to equipment in which no hardware changes have been made from the originally approved device to eliminate ambiguity about which hardware and software combinations have been approved.*" The proposed wording for the actual rules (paragraph (a) of **§ 2.1043 Changes in certificated equipment.**) seems to qualify paragraph 26, by suggesting that hardware changes, which do not affect transmitter characteristics, are permitted.

The first scenario involves simultaneous changes to both the hardware, and the software, of a radio that alters the characteristics of the transmitter. As an example, consider the addition of a memory module to a radio, which enables the execution of a new software load. The statement in the NPRM, paragraph 26, would disallow the use of the Class III permissive change, for this case. Motorola believes that this exclusion would be inconsistent with the intent of the Class III permissive change rule. We recommend that the rules more clearly differentiate between hardware that degrades emissions and safety, and hardware that does not.

[7] *See* Motorola comments at 33.

[8] *See NPRM* at para. 26.

The second scenario involves a software change to a device that has undergone a previous hardware change authorized by a Class II permissive change. Once again, Motorola sees no reason why this scenario presents any greater risk, in comparison to the case when no hardware change has been made. In both cases, the hardware-software combination will have undergone the same level of quality assurance (see Section 4). We recommend that the rules permit a Class III permissive change to apply to a radio that has previously undergone a Class II permissive change.

6.3. Class II Permissive Change and Software Changes

We see two issues pertaining to the definition, and application, of a Class II permissive change. First, the NPRM is somewhat inconsistent in its definition of a Class II permissive change when it states:[9] *"Class II permissive changes include modifications other than frequency, modulation or power that affect the RF emissions from a device."* On the other hand, the actual rules (paragraph (2) of **§ 2.1043 Changes in certificated equipment.**) state: *A Class II permissive change includes those modifications which degrade the performance characteristics as reported to the Commission at the time of the initial certification.* The inconsistency between these two statements (one statement uses "affect", the other, "degrade") should be corrected to reflect the rule's clear meaning that degradation of performance triggers a Class II permissive change.

The second issue centers around the authorization of a software change, which does not affect frequency, modulation, or power, but does degrade the undesired emissions from the radio. The NPRM does not clearly indicate whether the Class II, or Class III permissive change rule should be applied to this case. Motorola interprets the

NPRM to imply that the Class II permissive change should apply.

6.4. Authorization Process Matrix

In order to clarify the interpretation of the rules, it may be advantageous to create a matrix, such as those presented in the following sections. The matrices indicate which equipment authorization process is required, as a function of various relevant conditions. The first matrix interprets the rules as currently put forth in the NPRM. The second, simplified, matrix reflects Motorola's proposed modification to the rules. Our proposal is to incorporate several recommendations made earlier in this response. Those recommendations are:

1. Allow Class III permissive changes to be made to any capable radio, without having required an original declaration that the radio is a SDR.
2. A Class III permissive change may reflect concurrent changes to both software and hardware, when the hardware does not degrade desired, or increase undesired, emissions.
3. Allow Class III permissive changes to be made to a radio that has previously undergone a Class II permissive change.

For completeness, the following matrices include software changes made by third parties. Motorola sees significant logistical issues associated with third party software changes that affect emissions, which are discussed in Section 7.

The following definitions apply to the Authorization Process Matrices shown in the following sections.

- Software Change Type 1: A change to software, which does not alter

[9] *Id*. at para. 23.

frequency, modulation, or output power, and does not degrade undesired emissions from a transmitter.

- Software Change Type 2: A change to software, which does not alter frequency, modulation, or output power, but does degrade undesired emissions from a transmitter to a level that is still compliant with applicable rules.
- Software Change Type 3: A change to software that does alter frequency, modulation, or output power from a transmitter.
- Concurrent or Previous Hardware Change: A concurrent change to hardware which is not associated with the generation, control and amplification of the desired emissions, or, a previous hardware change authorized as a Class II permissive change.

6.5. Authorization Process Matrix 1: Interpretation of NPRM

The following matrix interprets the authorization rules as currently put forth in the NPRM. Refer to Section 6.4 for a detailed explanation.

	No Concurrent or Previous Hardware Change				Concurrent or Previous Hardware Change			
Declared Device SDR?	No		Yes		No		Yes	
Source of SW change?	Original grantee	Third party	Original grantee	Third party	Original grantee	Third party	Original grantee	Third party
SW Change Type 1	Class I	Class I	Class I	Class I	Class I	Class I	Class I	Class I
SW Change Type 2	Class II	N/A	Class II	New Filing	Class II	N/A	Class II	New Filing
SW Change Type 3	New Filing	N/A	Class III	New Filing	New Filing	N/A	New Filing	New Filing

6.6. Authorization Process Matrix 2: Proposed Modification to NPRM

The following matrix interprets the authorization based on recommendations summarized in Section 6.4.

Source of SW change?	Original grantee	Third party
SW Change Type 1	Class I	Class I
SW Change Type 2	Class II	New filing
SW Change Type 3	Class III	New filing

6.7. Additional Comments re NPRM Paragraph 28

The following section of the NPRM poses several specific questions pertaining to equipment authorization.[10] *"In addition, we seek comments on whether this new class of permissive change should be limited to software changes only, whether we should allow a combination of hardware and software permissive changes in a single device, whether there is a need for applicants to submit a copy of radio software to the Commission, and whether we should place limits on the number of hardware and software combinations under a single approval. We further seek comment on the benefits of the proposed new permissive change compared to the existing requirement for new identification numbers if we allow the alternative labeling method described below."* Motorola has already commented on the relationship between the Class III permissive change rule and hardware changes (Sections 6.2 and 6.3).

Motorola does not believe that it would be beneficial for applicants to submit a copy of radio software, as part of the equipment authorization process. It would, however, be reasonable for applicants to submit certain basic software identification information, such as software version numbers, and software revision dates. As

[10] *See NPRM* at para. 28.

discussed elsewhere in this response, Motorola believes that equipment manufacturers have the responsibility to insure that only approved hardware-software combinations can operate in Software Defined Radios.

Regarding potential limitations on the permissible number of combinations under a single approval, Motorola does not recommend that such a limitation be established. A limitation would tend to inhibit technology that enables common hardware platforms that support a large variety of software configurations.

Motorola addresses the question of third party software changes, and electronic labeling, in the following section.

7. Third Party Software Changes

7.1. General Comments on Third Party Software Changes

Motorola encourages, and is an active participant in, the emerging, vigorous and competitive third-party software market. The software products that define this market are primarily applications that execute within secure, trusted domains, like WAP or MExE. Motorola does not envision, in the foreseeable future, a significant extension of this market to include third party software products that directly control core radio functionality.

Market demands for low cost, small, power efficient products dictate the need for highly integrated solutions. Manufacturers will continue to exploit SDR technologies to enable greater and greater flexibility within these highly integrated designs. However, the added burden of designing products which accommodate lower layer software components (i.e. those which directly affect the radio subsystem), independently developed by third parties, would potentially add cost and size to radios which the market will not tolerate. Security, including concerns over interference and safety, is one of the main issues associated with this discussion. The topic of security is specifically addressed in Section 8 of this response.

In spite of our stated reservations, Motorola appreciates the FCC's desires to create a regulatory environment that encourages the continued emergence of third party software markets. We offer, therefore, in the following paragraphs, additional comments on this subject.

7.2. Electronic Labeling

Motorola endorses the FCC suggestion of electronic labeling.   With the creation of the Class III permissive change, the need for electronic labeling would be limited to software changes performed by third parties. We recommend, however, that electronic labeling be made optional for all authorizations, including original grants. Electronic labeling would simplify the process of delivering new products to market, and would allow for more information to be accessed than that which can be practically printed on a physical label.

We believe that all electronic labeling should be optional, so as to not place unnecessary burden on certain types of radio equipment, particularly equipment which does not include an electronic display. We also believe that electronic labeling should be allowed to take the form of either a visible display device (e.g. LCD, LED or similar alpha-numeric display capability) or an alternative means of extracting the electronic labeling information from the radio device (e.g. a terminal which can communicate with the radio device through any appropriate means.)

7.3. Administration Problems Associated with Third Party Software Changes

Independent from the concerns discussed in Section 7.1 and Section 8., Motorola sees several logistical complications associated with the proposed rules governing third party software changes. These concerns are about anticipated problems that the FCC, equipment manufacturers, and third party software vendors may face, as they seek to apply the rules proposed in the NPRM. In addition, these same issues have indirect implications to interference and safety. These problems will be described as scenarios in the following sections.

7.4. Administration Problem: OEM Makes a Change to Hardware Platform

An Original Equipment Manufacturer (OEM) is granted authorization for a radio. Following that, a third party software vendor obtains a new authorization for the combination of his software, and the original hardware. Following that, the OEM makes a Class II permissive hardware change to the radio. **Questions**: How will it be insured that the combination of new hardware and third party software operate in compliance with regulations? What authorization is required? Who has responsibility to obtain this authorization? How will it be insured that users do not use this combination if the combination is not compliant? A variation of this scenario could involve a completely new radio, authorized through a new filing.

7.5. Administration Problem: OEM Makes a Change to Software Platform

An Original Equipment Manufacturer (OEM) is granted authorization for a radio. Following that, a third party software vendor obtains a new authorization for the combination of his software, and the original hardware (which includes an operating system, and some embedded radio software). Following that, the OEM makes a Class II permissive software change to the radio, based on a change to the embedded radio software. **Questions**: How will it be insured that the combination of new software and third party software operate in compliance with regulations? What authorization is required? Who has responsibility to obtain this authorization? How will it be insured that users do not use this combination, if the combination is not compliant? A variation of this scenario could involve the OEM making a Class I permissive change (e.g. a change to the operating system).

7.6. Administration Problem: Original Device does not Transmit as Sold

An Original Equipment Manufacturer (OEM) markets a computing device, which contains embedded radio resources, but does not include software that enables any radio functionality to the end user. Consequently, the device, as sold, cannot act as a transmitter. Following that, a third party markets software, which when used in combination with the device, allows the original device to operate as a radio. **Questions**: What are the equipment authorization requirements for the OEM? How would this case affect the situation in the previous two scenarios?

7.7. Administration Problem: Third Party Grantee Makes Software Change

An Original Equipment Manufacturer (OEM) is granted authorization for a radio. Following that, a third party software vendor obtains a new authorization for the combination of his software, and the original hardware. Following that, the third party makes a software change that does not affect frequency, modulation, or output power, but does degrade undesired emissions to a level that is still compliant with applicable specifications. **Questions**: What are the authorization requirements for the new grantee (the third party)? The NPRM seems to imply that a completely new filing is required.

7.8. Recommendations to Address Administration Problems

Motorola sees significant issues surrounding the third party scenarios discussed in the preceding sections. In particular, we see the potential for serious breaches to regulatory controls, associated with the scenarios described in Sections 7.4 and 7.5. We recommend, therefore, that the FCC consider instituting some type of joint authorization for third party software changes, which would hold both the OEM and the third party, jointly accountable for the safe and reliable operation of the hardware-software combination.

8. Security

8.1. Open Interfaces and Security

As discussed in our NOI response, Motorola supports open interfaces at the application layer and encourages the emerging vigorous and competitive third-party software market.[11] Motorola believes that lower-layer software interfaces should remain under the control of the equipment manufacturer. This refers to interfaces that directly affect the radio subsystem. This position will insure the radios employing SDR technologies will operate reliably, and safely, and will not cause interference with other radios and services. Robust security methods are essential to insure that these important considerations are not compromised. Security is ultimately the responsibility of equipment manufacturers to ensure that their products are reliable and tamper-proof. The following sections outline the process by which equipment manufacturers can insure that SDR does not compromise security. (To aid in the discussion of security, Appendix A provides a glossary of commonly used security terminology.)

8.2. Elements of Security

The continued emergence of SDR technologies, and the services which they enable, will heighten the need for effective security in commercial wireless systems. The controlled environments in which commercial base stations operate provide greater inherent security, in comparison to commercial handsets. This point is underscored by the fact that second generation (2G) commercial base stations are remotely programmable, and have been operating in high volume for ten years without any significant security issues. The focus of this discussion, therefore, will be on security issues surrounding commercial handsets. Security requirements can be divided into the following five general categories:

Trusted System Operation: Confidence that software will execute in the device exactly as intended.

Authentication: refers to the ability to validate the origin of received information.

Integrity: assurance that received information has not been modified in transit.

Privacy: assurance that confidential information cannot be accessed by others.

Non-repudiation: positive verification of a sender's participation in a transaction.

[11] *See* Motorola Comments at 34.

Realization of the first requirement, Trusted System Operation, is achieved primarily through product design; it requires methodical architecture and domain analysis of the microprocessor systems within the device. The last four categories imply both requirements within the device, as well as the use of a robust security system framework, such as Public Key Cryptography. In the following paragraphs, these requirements are explored in more detail.

8.3. Important Security Considerations for Handheld Devices

A fundamental principle for designing a secure handset is the assumption that all design information, including the software source code, is available to the attacker. It should be assumed that the only information that is not available to the attacker are the private encryption keys that are securely kept either by the equipment manufacturer, by the network operator, by a trusted Public Key Infrastructure (PKI) service provider, or internal to the handset. Another important consideration when designing a secure handset is the likelihood that an attack method, developed by a sophisticated hacker, will be made available to a large number of users, possibly via the Internet. For example, a method to increase transmitter output power could become widely distributed as a PC program that accesses a handset through its test port. The handset design should prevent attacks that could easily be implemented by a large number of users. This makes securing the test port, keypad entry, and SIM interface, essential.

8.4. Principles of Public Key Cryptography

The principles of Public Key Cryptography (which is widely adopted within the Internet world) will not be addressed in detail here. Treatment of this subject is available from numerous sources. As a brief overview, the following four diagrams illustrate the generic process of Public Key Cryptography.

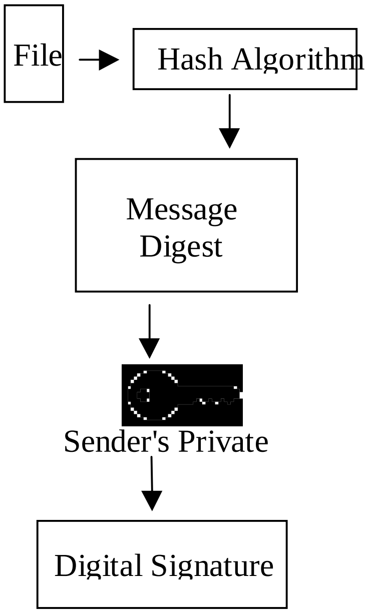

Figure 1: Digital Signature

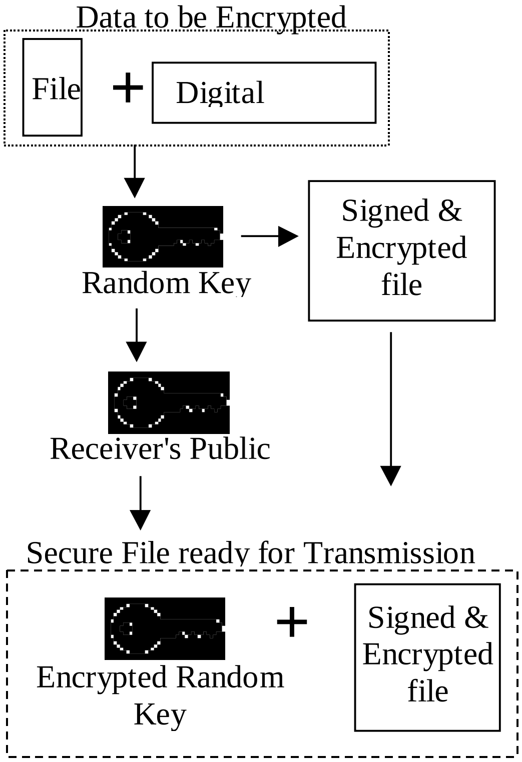

Figure 2: Encryption

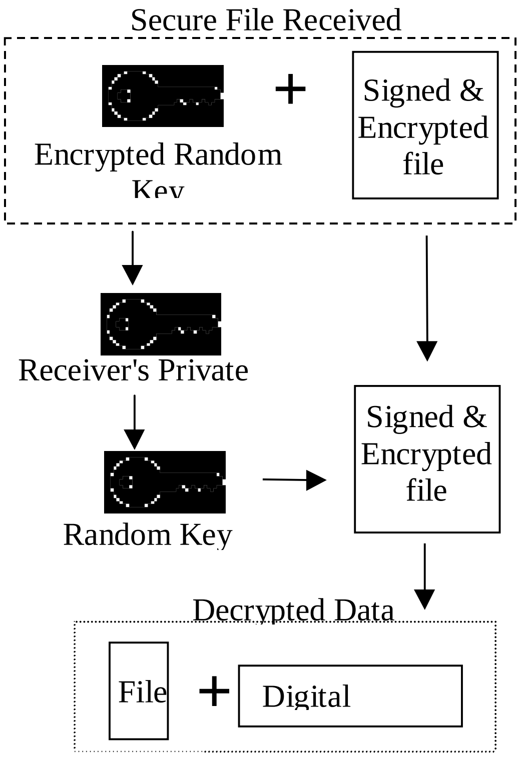

Figure 3: Decryption

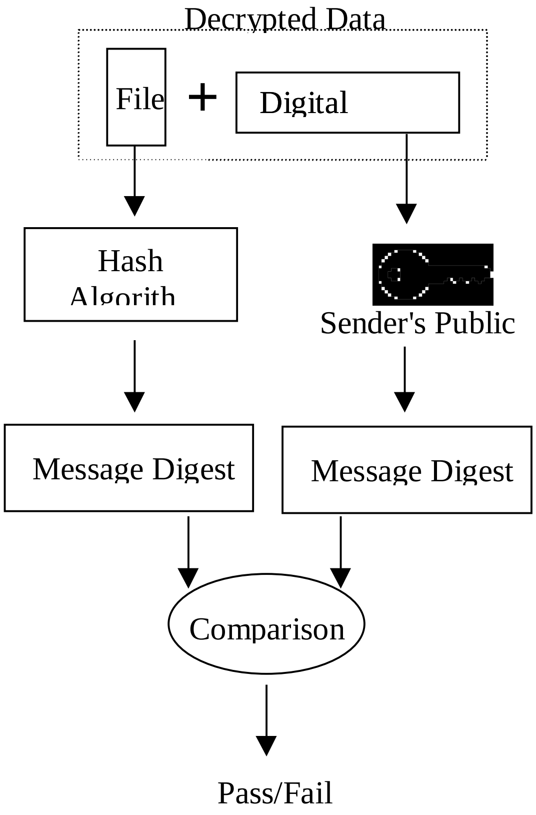

Figure 4: Verification

8.5. Encryption and Verification using Public and Private Keys

By basing security on private keys that can be securely stored, a handset can be designed that is secure against a sophisticated array of attacks, without adding significant cost to the device. Private keys will be either Rivest-Shamir-Adleman (RSA) or elliptic-curve keys. In RSA and elliptic-curve cryptographic systems, each private key has a corresponding public key. Public keys are stored at the equipment that will be receiving the software that must be decrypted, or have the signature verified before allowing it to operate. Verification assures that the software was properly signed and has not been modified. A typical implementation will utilize a Hashing function, such as the Secure Hash Algorithm (SHA-1), and a cryptography set of functions such as RSA. The SHA-1 hash function is used to map a large software data file to a 160-bit data block. The private RSA key is then used to create a signature of the data block that is a unique representation of the private key and the hashed block. The receiving side can use the public key to recreate the original 160-bit data block and check it against a locally generated block. By using a small RSA exponent, the verification can be implemented in software and can take less than a quarter of a second on a typical handset. The verification keys, and associated software, should, ideally, be implemented in tamper resistant hardware and software elements that cannot be modified by an attacker.

8.6. Compatibility with Existing Wireless Standards

It is reasonable to expect that equipment manufacturers will use security methods that are well established in the Internet, and are compatible with existing wireless standards such as the Wireless Application Protocol (WAP) and Mobile Execution Environment (MExE). These standards use Public Key Cryptography as the basic security mechanism. By using these standards as the foundation for SDR security, the cost impact of security can be minimized.

8.7. Security Design Requirements

Effective design for security begins with a well-articulated set of requirements.   It is necessary, therefore, to define detailed requirements, expressed in the form of security threat scenarios. For each scenario, a specific level of security robustness is specified. The compilation of all threat scenarios may take the form of the table presented below. For commercial handsets, employing SDR technologies, the threats can be structured into three categories: Device Configuration, Application Software Download, and Core Software Download.

Device Configuration (sometimes referred to as Provisioning) involves the download of relatively small files of non-executable software. These files contain data elements that will cause the device to reconfigure itself within a predefined range. Application Software is software that is implemented in a protected environment (called a "sandbox" in JAVA lexicon). Security for Application Software depends on containment security provided by a virtual machine or browser. Examples of protected application environments are J2ME (kJAVA), Personal JAVA, HMTL, and WML. Core Software, or Native Software, is software which does not run in a protected environment, and, therefore, could have unlimited access to data and resources on the device. Core Software includes software that controls RF emissions, frequency, modulation, and output power. Security requirements for Core Software are, therefore, much more demanding than the requirements for Application Software.

For each threat scenario, one of the following three security levels is assigned.

(For threats that involve malicious intent, it is assumed that the attacker has access to all design information and source code for the handset device.)

“Should not be easy” indicates that an average user cannot accomplish the attack based on instructions accessed over the Internet.

“Should not be practical” is defined as taking more than six months on equipment available today or costing more than $1000 per device by a well-equipped team of experts (Well equipped team of experts is defined below).

“Should not be feasible” is defined as taking more than 1,000 years or unlimited budget given the “cryptographic knowledge” that is available today by a well-equipped team of experts. “Cryptographic knowledge” does not include government-classified information.

“Well equipped team of experts” is defined as graduate level students with access to University equipment and collaborative access using the Internet. This level does not include a major Government Research Lab. It is assumed that once the attack has been developed, it will be published on the Internet.

NOTE: In the following table, certain product capabilities are indicated as "Allowed". These capabilities are shown so as to put the specified threats into clearer context.

No.	Requirement	Allowed	Easy	Practical	Feasible
1	**Device Configuration**				
1.1	Modification of the configuration of a device by any means except direct probing of ICs.			**X**	
1.2	Delegate configuration control to an operator or third party.	**X**			
2	**Application Software Download**				
2.1	Do not allow JAVA code (or other protected code) to access any processor memory outside of the designated environment.			**X**	
2.2	Do not allow JAVA code (or other protected code) to implement a Trojan horse that will access any data (key strokes, received data packets, etc.) outside the designated environment.			**X**	
2.3	Do not allow JAVA code (or other protected code) to present a user interface that remains active when the designated task (i.e. applet) is no longer active.			**X**	
2.4	Do not allow JAVA code (or other protected code) that will cause the basic subscriber unit to "crash".			**X**	
2.5	JAVA code (or other protected code) shall be allowed access to system critical resources only if there is an enforced multi-tier security plan (i.e. MExE) in place and the security separation is enforced.			**X**	

No.	**Requirement**	**Allowed**	**Easy**	**Practical**	**Feasible**
3	**Core Software Download**				
3.1	Allow only downloaded software with appropriate authentication to access sensitive security information (For example: Credit Card Number) in a subscriber product.				**X**
3.2	Do not allow software that will run native on subscriber products without the appropriate authentication.				**X**
3.3	It should be possible for manufacturer to delegate software download control to operators and 3rd party participants if desired.	**X**			
3.4	The user should not be able to claim that he did not receive a software downloaded feature. (Non-Repudiation Requirement).		**X**		
3.5	No single software "bug" should be able to cause transmission outside of the assigned spectrum.		**X**		
3.6	No single software "bug" should be able to cause the transmission to exceed the duty cycle, bandwidth limitation, or transmit power limits in a way that a subscriber unit will interfere with other users.		**X**		
3.7	Do not allow a virus that will cause transmission outside of the assigned spectrum.				**X**
3.8	Do not allow a virus that will cause improper program execution.				**X**
3.9	Do not allow modification of data in a Software Defined Radio that will cause interference to other services.				**X**

8.8. Security Implementation Guidelines

The following list summarizes implementation guidelines that can be followed in order to achieve the security requirements described in the Section 8.7. This list is not intended to be all encompassing, nor does it represent the only design approach that satisfies the security requirements. It is offered as an illustration of the type of guidelines that equipment manufacturers are following in the design of Software Defined Radios.

- Assume that all design information is available to the attacker

- Use RSA or elliptic curve cryptography with secure storage
- Design security that builds on the existing standards
- Use a security protocol that has been widely reviewed for security flaws; avoid the temptation to design a new protocol
- Use SHA-1 and RSA to verify test commands and software
- Include a tamper resistant equipment serial number
- Use public key based access codes to secure test ports
- Ensure that untrusted software runs in a protected environment so that it cannot access sensitive data or critical radio operations
- Do not assume that cellular over-the-air security is sufficient
- Ensure that the random number generator output is truly random
- Design software, including the operating system, to meet FIPS 140-1 level 2
- Software designed to access sensitive data and critical radio operations must be rigorously reviewed, verified and configuration managed
- Design software to be modular re-locatable and replaceable, so that software upgrades are feasible and reliable
- Engineers often put in backdoors for convenience (e.g., for testing, designing, debugging, etc.); these must be removed prior to commercial release
- Software that processes any kind of user input must have tight bounds checking against input buffer overrun attacks
- Erase or randomize all sensitive memory, including peripheral devices, so that a new program will not be able to access the previous memory contents
- Ensure that programs are properly terminated

- Ensure that all child processes are terminated when the parent task is terminated
- Ensure that child processes cannot inherit greater access to sensitive data or radio operations than the parent

9. Conclusion

Motorola appreciates the Commission's efforts to adjust its rules to realize the benefits of SDR for the American public. We hope that our comments herein will be helpful in assisting the Commission in streamlining its equipment authorization rules to reflect the additional flexibility such radios offer.

Respectfully Submitted,

_______/S/________
Richard C. Barth
Vice President and Director,
Telecommunications Strategy
and Regulation

_______/S/________
John F. Lyons
Director, Telecommunications
Strategy and Regulation

Motorola, Inc.
1350 I Street, N.W., Ste 400
Washington, DC 20005-3305
Tel: 202-371-6900

March 19, 2001

Appendix : Glossary of Security Terminology

Access Code: is a certificate used to prevent unauthorized programming of a subscriber device.

APCO25: a US based standard for secure dispatch systems.

Authentication: assures that the receiver of a message can ascertain its origin.

Authentication code: a cryptographic checksum based on an approved security function (also known as a Message Authentication Code (MAC) in ANSI standards).

Cookies: Small memory blocks that an HTML page uses to store information that can be accessed next time that HTML page is made active.

Compromise: the unauthorized disclosure, modification, substitution, or use of critical security parameters (including plain-text cryptographic keys and other CSPs).

Confidentiality: the property that critical security parameters are not disclosed to unauthorized individuals, entities, or processes.

Critical security parameter (CSP): security-related information (e.g., secret and private cryptographic keys, and authentication data such as passwords and PINs) appearing in plain-text or otherwise unprotected form and whose disclosure or modification can compromise the security of a cryptographic module or the security of the information protected by the module.

Cryptographic key (key): a parameter used in conjunction with a cryptographic algorithm that determines the transformation of plain-text data into cipher-text data, the transformation of cipher-text data into plain-text data, a digital signature computed from data, the verification of a digital signature computed from data, an authentication code computed from data, or an exchange agreement of a shared secret.

Differential power analysis (DPA): an analysis of variations of the electrical power consumption of a device, using advanced statistical methods and/or error correction techniques, for the purpose of extracting information correlated to encryption keys used in a cryptographic algorithm.

Digital Signature: a non-forgeable transformation of data that allows proof of the source (with non-repudiation) and verification of the integrity of that data.

Dongle: is a token device that must be present to access critical security parameters or features in a subscriber device. The dongle is usually plugged into a PC that is used to access or modify information on the subscriber unit.

ECC: Elliptic-Curve Cryptography is a type of Public Key Cryptography that is based on finding points on Elliptic Curves. An ECC key is smaller than an equivalently secure RSA key, and for most applications ECC operations are significantly faster.

Electromagnetic interference (EMI): electromagnetic phenomena that either directly or indirectly can contribute to degradation in the performance of an electronic system.

Firmware: the programs and data stored in hardware (e.g., ROM, PROM, or EPROM) such that the programs and data cannot be dynamically written or modified during execution.

Hardware: the physical equipment used to process programs and data in a cryptographic module.

Hash-based message authentication code (HMAC): a message authentication code that utilizes a keyed hash.
Initialization vector (IV): a vector used in defining the starting point of an encryption process within a cryptographic algorithm.
Input data: information that is entered into a cryptographic module for the purposes of transformation or computation. Integrity: the property that critical security parameters have not been modified or deleted in an unauthorized and undetected manner.
JTAG: a standard describing a test method using a boundary scan architecture.
Key encrypting key: a cryptographic key that is used for the encryption or decryption of other keys. Key management: the activities involving the handling of cryptographic keys and other related security parameters (e.g., IVs and passwords) during the entire life cycle of the keys, including their generation, storage, distribution, entry and use, and deletion or destruction.
Manual key distribution: a non-electronic means of distributing cryptographic keys.
Manual key entry: the entry of cryptographic keys into a cryptographic module using devices such as a keyboard.
Non-Repudiation: Uses cryptographic technology to assure that a sender cannot falsely deny that a transaction took place.
One-Way Hash Function: takes variable length input and converts it to a fixed-length output. This is done is a way that makes it very difficult to create a different input that creates the same output. SHA-1 is an example of a strong one-way hash function.
Password: a string of characters (letters, numbers, and other symbols) used to authenticate an identity or to verify access authorization.
Personal identification number (PIN): a 4 or more character alphanumeric code or password used to authenticate an identity (commonly used in banking applications).
Plain-text key: an unencrypted cryptographic key.
Private key: a cryptographic key, used with a public key cryptographic algorithm, which is uniquely associated with an entity and is not made public.
Public key: a cryptographic key, used with a public key cryptographic algorithm, uniquely associated with an entity, and that may be made public. (Public keys are not considered CSPs.)
Public key certificate: a set of data that unambiguously identifies an entity contains the entity’s public key, is digitally signed by a trusted party, binding the public key to the entity.
Public-Key Cryptography: a cryptographic algorithm that uses two related keys, a public key and a private key. The two keys have the property that, given the public key, it is computationally infeasible to derive the private key.
RSA: is a type of Public-Key Cryptography that is based on the difficulty of factoring large numbers.
Security Kernel: is software that controls access to critical security parameters. Tasks that require access to critical security parameters must request access from the security kernel. The security kernel uses the MMU to enforce memory separation.
Secret key: a cryptographic key, used with a secret key cryptographic algorithm that is uniquely associated with one or more entities and should not be made public. The use the term "secret" in this context does not imply a classification level; rather the term implies the need to protect the key from disclosure or substitution.

Secret key (symmetric) cryptographic algorithm: a cryptographic algorithm that uses a single secret key for both encryption and decryption.
Seed key: a secret value used to seed a cryptographic function or operation.
Should not be easy: indicates that an average user cannot do this based on instructions accessed over Internet. It is assumed that a "well-equipped team of experts" developed the attack.
Should not be practical: is defined as taking more than six months on equipment available today or costing more $1000 per device.
Should not be feasible: is defined as taking more than 1,000 years or unlimited budget given the "cryptographic knowledge" that is available today. "Cryptographic Knowledge" does not include Government Classified information.
SHA-1: is a widely used one-way hash algorithm. It convert a large file into a 160-bit hashed value.
Symmetric-Key Cryptography: uses the same key for decoding as for encoding. An attacker that knows how to decode a message also knows how to encode a message.
Simple power analysis (SPA): a direct (primarily visual) analysis of patterns of instruction execution (or execution of individual instructions), obtained through monitoring the variations in electrical power consumption of a device, for the purpose of revealing the features and implementations of cryptographic algorithms and subsequently the values of cryptographic keys.
Split knowledge: a condition under which two or more entities separately have key components that individually convey no knowledge of the plain-text key that will be produced when the key components are combined in the cryptographic module.
System software: the special software (e.g., operating system, compilers or utility programs) designed for a specific computer system or family of computer systems to facilitate the operation and maintenance of the computer system, programs, and data.
Tamper detection: the automatic determination by a cryptographic module that an attempt has been made to compromise its physical security
Well-equipped team of experts: is defined as graduate level students with access to university equipment and collaborative access using Internet. This level does not include a major Government Research Lab. It is assumed that once the attack has been developed, it will be published on the Internet.
Zeroization: a method of erasing electronically stored data by altering or deleting the contents of the data storage so as to prevent the recovery of the data.